Dependability Evaluation of Cluster-based Distributed Systems

Emmanuelle Anceaume, Francisco Brasiliero, Romaric Ludinard, Bruno Sericola, Frédéric Tronel

International Journal of Foundations of Computer Science

DEPENDABILITY EVALUATION OF CLUSTER-BASED DISTRIBUTED SYSTEMS*

EMMANUELLE ANCEAUME
CNRS / IRISA, Campus Universitaire de Beaulieu, Rennes, France
emmanuelle.anceaume@irisa.fr

FRANCISCO BRASILEIRO
Universidade Federal de Campina Grande, LSD Laboratory, Campina Grande, Brazil
fubica@dsc.ufcg.edu.br

ROMARIC LUDINARD,† BRUNO SERICOLA
INRIA Rennes Bretagne-Atlantique, Rennes, France
{*romaric.ludinard,bruno.sericola*}*@inria.fr*

FRÉDÉRIC TRONEL
Supélec, Rennes, France
ftronel@rennes.supelec.fr

Received (Day Month Year)
Accepted (Day Month Year)
Communicated by (xxxxxxxxxx)

Awerbuch and Scheideler have shown that peer-to-peer overlay networks can survive Byzantine attacks only if malicious nodes are not able to predict what will be the topology of the network for a given sequence of join and leave operations. In this paper we investigate adversarial strategies by following specific protocols. Our analysis demonstrates first that an adversary can very quickly subvert overlays based on distributed hash tables by simply never triggering leave operations. We then show that when all nodes (honest and malicious ones) are imposed on a limited lifetime, the system eventually reaches a stationary regime where the ratio of polluted clusters is bounded, independently from the initial amount of corruption in the system.

1. Introduction

The adoption of peer-to-peer overlay networks as a building block for architecting Internet scale systems has raised the attention of making these overlays resilient not only to benign crashes, but also to malicious failure models [1, 2, 3]. As a result,

*A shorter version of this paper appeared in the 2nd International Workshop on Reliability, Availability and Security (WRAS), February 2010.
†Supported by the Direction Générale des Entreprises - P2Pim@ges project

Byzantine-resilient overlays have been proposed (e.g., [4, 5, 6, 7]). Awerbuch and Scheideler [8] have shown that peer-to-peer overlay networks can survive Byzantine attacks only if malicious nodes are not able to predict what will be the topology of the network for a given sequence of join and leave operations. A prerequisite for this condition to hold is to guarantee that nodes identifiers randomness is continuously preserved. However this is not sufficient as by holding a logarithmic number of addresses the adversary can disconnect some target from the overlay in a linear number of trials. Similarly, randomly placing data in the network, as operated in unstructured overlay networks, guarantees that data are difficult to attack however requires a linear number of queries to be retrieved which is not scalable [9]. Inducing churn has been shown to be the other fundamental ingredient to preserve randomness. Churn is classically defined as the rate of turnover of peers in the system [10], and thus induced churn refers to the general idea of forcing peers to move within the system. Several strategies based on this principle have been proposed. Most of them are based on locally induced churn. However either they have been proven incorrect or they involve a level of complexity that is too high to be practically acceptable [8]. The other ones, based on globally induced churn, enforce limited lifetime for each node in the system. This keeps the system in an unnecessary hyper-activity, and thus needs to impose strict restrictions on nodes joining rate which limits its applicability to open systems.

In this paper we propose to leverage the power of clustering to design a practically usable solution that preserves randomness under an adaptive adversary. Our solution relies on the clusterized version of peer-to-peer overlays combined with a mechanism that allows the enforcement of limited nodes lifetime. Cluster-based overlays have revealed to be well adapted for reducing both the impact of churn on the system and the damage caused by failures in the absence of targeted attacks [6, 11, 4]. The contributions of this paper are two-fold. We first investigate adversarial strategies by following specific protocols. Our analysis demonstrates that an adversary can very quickly subvert cluster-based overlays by simply never triggering leave operations. We then show that when all nodes are imposed on a limited lifetime, the system eventually reaches a stationary regime where the ratio of polluted clusters is bounded.

The remainder of this paper is organized as follows: In Section 2 we briefly describe the main features of cluster-based overlays, and their dependability issues. Section 3 presents the overlay operations. In Section 4 the adversarial behavior is modeled and its impact at cluster level by using a Markovian analysis is studied. In this section, we consider a non restricted adversary. Section 5 is devoted to the same study in the case of a restricted adversary. Section 6 shows that by inducing churn at all peers of the system, safety of the system is preserved. Section 7 demonstrates the practicability of our solution. We conclude in Section 8.

2. Overlay Networks

An overlay network is a virtual network built on top of a physical network. Nodes of the overlay, usually called *peers*, communicate among each other along the edges of the overlay by using the communication primitives provided by the underlying network (e.g., IP network service). The algorithms that peers use to choose their neighbors and to route messages define the overlay topology. The topology of unstructured overlays conforms with random graphs, i.e., relationships among peers are mostly set according to a random process which makes joining and leaving operations constraint free. Data placement enjoy the same absence of constraints. Any data can be placed on any peer thereby imposing flooding or random walk techniques to retrieve them. Randomly placing data in the network guarantees that data is difficult to attack but requires a linear number of queries to be retrieved which is definitively not scalable [9]. This scalability issue can be circumvented at the price of strong restrictions on churn [12]. Structured overlays (also called Distributed Hash Tables (DHTs)) build their topology according to structured graphs (e.g., hypercube, torus). For most of them, the following principles hold: each peer is assigned a unique random identifier from an m-bit identifiers space. Identifiers (denoted IDs in the following) are derived by applying some standard cryptographic one-way hash function on the peers network address. The value of m (128 for the standard MD5 hash function) is large enough to make the probability of identifiers collision negligible. The identifier space is partitioned among all the peers of the overlay. Peers self-organize within the structured graph according to a distance function D based on peers IDs (e.g., two peers are neighbors if their IDs share some common prefix), plus possibly other criteria such as geographical distance. Each application-specific object, or data-item, is assigned a unique identifier, called *key*, selected from the same m-bit identifiers space. Each peer p owns a fraction of all the data items of the overlay. The mapping derives from the distance function D. In the following, we will use the term peer (or key) to refer to both the peer (or key) and its m-bit representation. Following the seminal work of Plaxton et al [13], diverse DHTs have been proposed (e.g., CAN [14], Chord [15], Pastry [16], Kademlia [17]). All these DHTs have been proven to be highly satisfactory in terms of efficiency and scalability (i.e., their key-based routing interface guarantees operations whose complexity in messages and latency usually scale logarithmically with system size). However, in presence of adversarial behavior, relying on single peers to ensure the system connectivity and the correct retrieval of data is clearly not sufficient, as by holding a logarithmic number of addresses the adversary can in a linear number of trials disconnect some target from the overlay. On the contrary, by having peers gathered into quorums or clusters, one can introduce the unpredictability required to deal with Byzantine attacks through randomized algorithms. This has led to cluster-based structured overlay networks.

2.1. *Cluster-based Structured Overlay Networks*

Cluster-based structured overlay networks are such that clusters of peers substitute for peers at the vertices of the structured graph. Each vertex of the structured graph is composed of a set or *cluster* of peers. Peers join the clusters according to a given distance metric D. For instance in PeerCube [6], peer p joins the (unique) cluster whose label is a prefix of p's identifier, while in eQuus [11], p joins the (unique) cluster whose members are geographically the closest to p. Clusters in the overlay are uniquely labelled. Size of each cluster is both lower and upper bounded. The lower bound, named c in the following, usually satisfies some constraint based on the assumed failure model. For instance $c \geq 4$ allows Byzantine tolerant agreement protocols to be run among these c peers despite the presence of one Byzantine peer [18]. The upper bound, that we call S_{max}, is typically in $\mathcal{O}(\log U)$ where U is the current number of peers in the overlay, to meet scalability requirements. Once a cluster size exceeds S_{max}, this cluster `splits` into two smaller clusters, each one populated with the peers that are closer to each other according to distance D. Once a cluster undershoots its minimal size c, this cluster `merges` with the closest cluster in its neighborhood. For space reasons we do not give any detail regarding localization of a cluster in the overlay nor the `split/merge` operations. None of these processes are strictly necessary to understand our work. The interested reader is invited to read their descriptions in the original papers (e.g. [4, 6, 11]).

In the present work we assume that at cluster level, peers are organized as core and spare members. Members of the core set are primarily responsible for handling messages routing and clusters operations. Management of the core set is such that its size is maintained to constant c. Spare members are the complement number of peers in the cluster. Size s of the spare set is such that $s \leq S$ where $S = S_{max} - c$. In contrast to core members, spare members are not involved in any of the overlay operations. Rationale of this classification is two-fold: first it allows to introduce the unpredictability required to deal with Byzantine attacks through a randomized core set generation algorithm as shown in the sequel. Second it limits the management overhead caused by the natural churn present in typical overlay networks through the spare set management. In the following we assume that join and leave events have an equal chance to occur in any cluster.

2.2. *Dependability Issues*

A fundamental issue faced by any practical open system is the inevitable presence of peers that try to manipulate the system by exhibiting undesirable behaviors [3]. Such peers are classically called malicious or Byzantine. Malicious peers can devise complex strategies to prevent peers from discovering the correct mapping between peers and data keys. They can mount *Sybil attacks* [19] (i.e., an attacker generates numerous fake peers to pollute the system), they can do *routing-table poisoning* (also called *eclipse attacks* [1, 3]) by having honest peers redirecting outgoing links towards malicious ones, or they can simply drop or re-route messages towards other

malicious peers. They can magnify their impact by colluding and coordinating their behavior. We model these strategies through a strong adversary that controls these malicious peers. We assume that the adversary has large but bounded resources in that it cannot control more than a fraction μ ($0 < \mu < 1$) of malicious peers in the whole network. Note that in the following we make a difference between the whole network and the overlay. The network encompasses all the peers that at some point may participate to the overlay (i.e. 2^m peers), while the overlay contains at any time the subset U of participating peers. Thus, while μ represents the assumed fraction of malicious peers in the network, the goal of the adversary is to populate some parts of the overlay with a larger fraction of malicious peers in order to subvert the correct functioning of the overlay. Finally, a peer which always follows the prescribed protocols is said *honest*. Note that honest peers cannot *a priori* distinguish honest peers from malicious ones.

2.3. *Security Schemes*

We assume the existence of a public key cryptography scheme that allows each peer to verify the signature of each other peer. We also assume that correct peers never reveal their private keys. Peers IDs and keys (private and public) are acquired via a registration authority. When describing the protocols, we ignore the fact that messages are signed, and recipients of a message ignore any message that is not signed properly. We also use cryptographic techniques to prevent a malicious peer from observing or unnoticeably modifying a message sent by a correct peer. However a malicious peer has complete control over the messages it sends and receives.

3. Operations of the Overlay

When a peer joins a cluster or leaves it, corresponding operations are executed. Design of these operations takes advantage of peers role separation. The `leave` operation for peers in the core set introduces a certain amount of unpredictability required to deal with Byzantine attacks through a randomized core set generation algorithm. On the other hand both `leave` operations for peers in the spare set and `join` operations have no impact on the overall topology (provided that the size of the concerned cluster does not reach its lower or upper bounds) discouraging brute force denial of service attacks. Specifically:

· **join(p):** when peer p `joins` the system, it joins the spare set of the closest cluster $\mathcal{G}$ in the system (according to distance D). Core member of $\mathcal{G}$ update their spare view to reflect p's insertion (note that the spare view update does not need to be tightly synchronized among all core members).

· **leave(p):** when peer p `leaves` its cluster $\mathcal{G}$, either p belongs to $\mathcal{G}$ spare view or to $\mathcal{G}$ core view. In the former case, core members of $\mathcal{G}$ simply update their spare view to reflect p's departure, while in the latter case the core view maintenance procedure is triggered. Two maintenance procedures are analyzed. The first one, called $procedure_1$, simply consists in replacing the left core member by one randomly

chosen among the spare members. The second one, called *procedure*$_2$, consists in renewing the whole core set by choosing c random peers among the whole cluster.

The `join` and `leave` operations make up the cluster protocol. In the following `Protocol`$_1$ refers to a `leave`(.) operation followed by a `join`(.) operation with *procedure*$_1$ as maintenance procedure. Similarly, `Protocol`$_2$ refers to a `leave`(.) operation followed by a `join`(.) operation using *procedure*$_2$.

4. Modeling the Adversarial Strategy in a Cluster

In this section, we investigate to which extent the two previously described protocols, i.e., `Protocol`$_1$ and `Protocol`$_2$, prevent the adversary from elaborating deterministic strategies to compromise the correctness of a targeted cluster $\mathcal{G}$. Correctness of a cluster is compromised as soon as a quorum of core members are malicious. Specifically, in both maintenance procedures, core members trigger a Byzantine-resilient consensus algorithm to agree on the composition of the renewed core view. It is well-known that a necessary and sufficient condition to prevent agreement among a set of nodes is that strictly more than a third of these nodes are malicious [18]. In our context, cluster $\mathcal{G}$ is said *polluted* if its core set is populated by strictly more than $\lfloor (c-1)/3 \rfloor$ malicious peers, with c the size of the core set. In the following, we denote by c' the value $\lfloor (c-1)/3 \rfloor$.

Both `Protocol`$_1$ and `Protocol`$_2$ are modeled by using the ball and urn model as presented in Figure 1. Specifically, the very large number of peers in the network is depicted by a potentially infinite number of white and red balls in a bag. White balls represent honest peers, while red ones represent malicious peers. There is a proportion $1-\mu$ of white balls and a proportion μ of red balls in the bag. In addition to the bag, we consider two urns, referred to as $\mathcal{C}$ and $\mathcal{S}$ in the following, which respectively represent the core set and the spare set of a cluster. Join and leave operations are modeled through the throwing and withdrawing of balls into and from the urns. For instance, drawing a ball from the bag and throwing this ball into $\mathcal{S}$ models the insertion of a new peer into the overlay and its joining into the spare set of a cluster. Drawing a ball from $\mathcal{S} \cup \mathcal{C}$ and throwing this ball into the bag models the leaving of a peer from the overlay. In Figure 1, both protocols are made of two indivisible tasks. Task 1 models the random choice of a peer within a cluster while Task 2 is concerned with the leave and join operations. We consider a succession of rounds $r_1, r_2, \ldots$ during which the rules of the protocols are applied. Rules are oblivious to the color of the balls, meaning that they cannot distinguish between the white and the red balls.

As previously said the goal of the adversary is to maximize the number of red balls in both urns $\mathcal{C}$ and $\mathcal{S}$ so that the number of red balls in $\mathcal{C}$ is bound to always exceed quorum $c' = \lfloor (c-1)/3 \rfloor$. By construction, the adversary has only access to red balls. Thus an effective strategy for the adversary to gain this quorum is to prevent red balls from being extracted from both urns. Specifically, in Task 1 of both protocols (see Figure 1), if the drawn ball b_0 is red then this ball is put back

/* Protocol$_1$*/
/* Task 1 */
draw a ball b_0 from $\mathcal{C} \cup \mathcal{S}$
/* Task 2 */
if $b_0 \in \mathcal{C}$ **then**
  draw a ball b_1 from $\mathcal{S}$
  throw b_1 into $\mathcal{C}$
endif
throw b_0 into the bag
draw a ball b_2 from the bag
throw b_2 into $\mathcal{S}$

/* Protocol$_2$*/
/* Task 1 */
draw a ball b_0 from $\mathcal{C} \cup \mathcal{S}$
/* Task 2 */
if $b_0 \in \mathcal{C}$ **then**
  throw all the balls of $\mathcal{C}$ into $\mathcal{S}$
  draw c balls from $\mathcal{S}$
  and throw these c balls into $\mathcal{C}$
endif
throw b_0 into the bag
draw a ball b_2 from the bag
throw b_2 into $\mathcal{S}$

Fig. 1. Rules of Protocol$_1$ and Protocol$_2$.

into the urn it was drawn from (i.e., either $\mathcal{C}$ or $\mathcal{S}$). In that case, Task 2 is not applied, and a new round is triggered. This strategy ensures that the number of red balls in $\mathcal{C} \cup \mathcal{S}$ is non decreasing.

We model the effects of rounds $r_1, r_2, \ldots$ using a homogeneous Markov chain denoted by $X = \{X_n, n \geq 0\}$ representing the evolution of the number of red balls in both urns $\mathcal{C}$ and $\mathcal{S}$. More formally, the state space Ω of X is defined by $\Omega = \{(x, y) \mid 0 \leq x \leq c,\ 0 \leq y \leq s\}$, and for $n \geq 1$ the event $X_n = (x, y)$ means that after the n-th transition or n-th round the number of red balls within urn $\mathcal{C}$ is equal to x and the number of red balls within urn $\mathcal{S}$ is equal to y. The transition probability matrix P of X depends on both the rules of the given protocol and the adversarial behavior. This matrix is detailed in each of the following subsections. We define a state as *polluted* if in this state urn $\mathcal{C}$ contains strictly more than $c' = \lfloor (c-1)/3 \rfloor$ balls. Conversely, a state that is not polluted is said to be *safe*. In the remaining of this paper, the initial probability distribution is denoted by α.

4.1. *Protocol$_1$*

Regarding Protocol$_1$, the subset of safe states A is defined as $A = \{(x, y) \mid 0 \leq x \leq c',\ 0 \leq y \leq s\}$, while the set of polluted states B, is the subset $\Omega - A$, i.e., $B = \{(x, y) \mid c' < x \leq c,\ 0 \leq y \leq s\}$. By the rules of Protocol$_1$, one can never escape from the subset of states B to switch back to safe states A since the number of red balls in $\mathcal{C}$ is non decreasing. Thus, the adversary wins the protocol when process X reaches the closed subset B, as illustrated in Figure 2. Matrix P and the initial probability vector α are partitioned in a manner that matches the decomposition of $\Omega = A \cup B$, that is

$$P = \begin{pmatrix} P_A & P_{AB} \\ 0 & P_B \end{pmatrix} \text{ and } \alpha = (\alpha_A \ \ \alpha_B),$$

where P_A (resp. P_B) is the sub-matrix of dimension $|A| \times |A|$ (resp. $|B| \times |B|$), containing the transitions between states of A (resp. B); P_{AB} is the sub-matrix of

dimension $|A| \times |B|$, containing the transitions from states of A to states of B; B is an absorbing class thus $P_{BA} = 0$. Finally, sub-vector α_A (resp. α_B) contains the initial probabilities of states of A (resp. B).

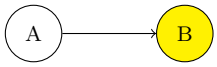


Fig. 2. Aggregated view of the Markov chains associated with $\texttt{Protocol}_1$. Safe states are represented by A, and polluted ones by B. B is an absorbing class.

The computation of the transition probabilities of matrix P is illustrated in Figure 3. In this tree, each edge is labelled by a probability and its corresponding event according to the rules of $\texttt{Protocol}_1$. This figure can be interpreted as follows: At round r, $r \geq 1$, starting from state (x, y) (root of the tree), the Markov chain can visit four different states, namely (x, y), $(x, y+1)$, $(x+1, y-1)$, and $(x+1, y)$ (leaves of the tree). The probability associated with each one of these transitions is obtained by summing the products of the probabilities discovered along each path starting from the root to the leaf corresponding to the target state. Derivation of the transition probability matrix P of Markov chain X associated with $\texttt{Protocol}_1$ is as follows: for all $x \in \{0, \ldots, c\}$ and for all $y \in \{0, \ldots, s\}$, we have

$$
\begin{aligned}
p_{(x,y),(x,y)} &= \left(\frac{1}{c+s}\right)\left(x + \frac{(c-x)(s-y)(1-\mu)}{s} + y + (s-y)(1-\mu)\right) \\
p_{(x,y),(x,y+1)} &= \frac{(c+s-x)(s-y)\mu}{(c+s)s} \quad \text{for } y \leq s-1 \\
p_{(x,y),(x+1,y-1)} &= \frac{(c-x)y(1-\mu)}{(c+s)s} \quad \text{for } x \leq c-1 \text{ and } y \geq 1 \\
p_{(x,y),(x+1,y)} &= \frac{(c-x)y\mu}{(c+s)s} \quad \text{for } x \leq c-1.
\end{aligned}
$$

In all other cases, transition probabilities are null.

4.2. *Protocol*$_2$

For $s = 1$, it is easy to see that $\texttt{Protocol}_2$ is equivalent to $\texttt{Protocol}_1$. On the other hand, in contrast to $\texttt{Protocol}_1$, $\texttt{Protocol}_2$ alternates between safe and polluted states. After a random number of these alternations the process enters a set of closed polluted states. Indeed, by the rules of the protocol, one can escape finitely often from polluted states (x, y) to switch back to safe states as long as (x, y) satisfies $x + y < s + c'$ (there are still sufficiently many white balls in both $\mathcal{C}$ and $\mathcal{S}$ so as to successfully withdraw c balls such that $\mathcal{C}$ is safe). However, there always exists a round such that a state (x, y) is entered, where $x + y \geq s + c'$. From this round onwards, going back to safe states is impossible: the adversary has won the protocol. Formally, the subset of safe states A is defined, as for $\texttt{Protocol}_1$, by $A = \{(x, y) \mid 0 \leq x \leq c',\ 0 \leq y \leq s\}$. On the other hand we need to decompose

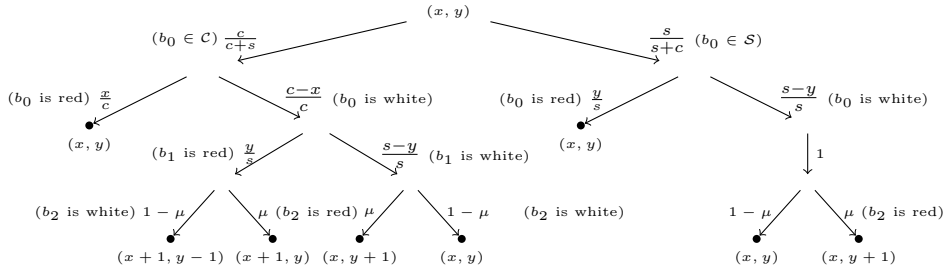


Fig. 3. Transition diagram for the computation of matrix P of $\texttt{Protocol}_1$.

the set B of polluted states into two subsets C and D defined by $C = \{(x, y) \mid x > c',\ x + y < s + c',\ 0 \leq y \leq s\}$, and $D = \{(x, y) \mid x > c',\ x + y \geq s + c',\ 0 \leq y \leq s\}$. When $s = 1$, we have $B = D$ and $C = \emptyset$. Subsets A and C are transient and subset D is a closed subset as illustrated in Figure 4.

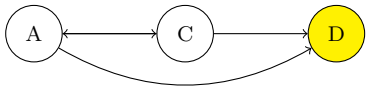


Fig. 4. Aggregated view of the Markov chains associated with $\texttt{Protocol}_2$. Safe states are represented by A, and polluted states by C and D. State D is an absorbing class.

Following the decomposition of $\Omega = A \cup C \cup D$, we partition matrix P and the initial probability vector α by writing

$$P = \begin{pmatrix} P_A & P_{AC} & P_{AD} \\ P_{CA} & P_C & P_{CD} \\ 0 & 0 & P_D \end{pmatrix} \text{ and } \alpha = (\alpha_A \ \ \alpha_C \ \ \alpha_D).$$

By proceeding similarly as in Section 4.1, we derive the transitions of process X associated with $\texttt{Protocol}_2$. Briefly, when the protocol starts in state (x, y) at round r, it remains in state (x, y) during the round if either ball b_0 is red or b_0 is white, it has been drawn from $\mathcal{S}$, and b_2 is white. It changes to state $(x, y+1)$ if b_0 is white, it has been drawn from $\mathcal{S}$, and b_2 is red. Finally the protocol switches to state $(k, x+y-k+\ell)$, where k is an integer $k = 0, \ldots, \min(c, x+y)$ and $\ell = 0, 1$ if b_0 is white and it has been drawn from $\mathcal{C}$, and the maintenance procedure leads to the choice of k red balls. For all $x \in \{0, \ldots, c\}$, $y \in \{0, \ldots, s\}$, and $s > 1$, we have

$$\begin{aligned} p_{(x,y),(x,y)} &= \frac{x}{c+s} + \frac{y}{c+s} + \frac{s-y}{c+s}(1-\mu) + \frac{c-x}{c+s}(1-\mu)q(x, x+y) \\ p_{(x,y),(x,y+1)} &= \frac{s-y}{c+s}\mu + \frac{c-x}{c+s}\mu q(x, x+y) \text{ for } y \leq s-1 \\ p_{(x,y),(k,x+y-k)} &= \left(\frac{c-x}{c+s}\right)(1-\mu)\, q(k, x+y) \text{ for } 0 \leq k \leq \min(c, x+y) \text{ and } k \neq x \end{aligned}$$

$$p_{(x,y),(k,x+y-k+1)} = \left(\frac{c-x}{c+s}\right) \mu q(k, x+y) \text{ for } 0 \leq k \leq \min(c, x+y) \text{ and } k \neq x$$

where

$$q(z, x+y) = \frac{\binom{x+y}{z}\binom{c+s-1-(x+y)}{c-z}}{\binom{c+s-1}{c}} 1_{\{0 \leq x+y-z \leq s-1\}} \tag{1}$$

is the probability of getting z red balls when c balls are drawn, without replacement, in an urn containing $x+y$ red balls and $c+s-1-(x+y)$ white balls, referred to as the hypergeometric distribution. $1_{\{\ldots\}}$ represents the indicator function. In all other cases, transition probabilities are null.

4.3. *Comparison of both Protocols in a Non Constrained Adversarial Environment*

As described in Section 4.1 the Markov chain X associated with $\texttt{Protocol}_1$ is reducible and the states of A are transient, which means that matrix $I - P_A$ is invertible, where I is the identity matrix of dimension $|A|$. Recall that B is an absorbing class, i.e. $P_{BA} = 0$. Similarly, as described in Section 4.2, Markov chain X associated with $\texttt{Protocol}_2$ is reducible, the states of A and C are transient and subset D is a closed subset.

4.3.1. *Initial Distributions*

In the experiments conducted for this work, we consider two initial distributions. The first one, which we denote by β, consists in drawing $c+s$ balls from the bag such that c of them are thrown into urn $\mathcal{C}$, and the other s ones are thrown into urn $\mathcal{S}$. This initial state X_0 is defined by $X_0 = (C_r, S_r)$ where C_r (resp. S_r) is the initial number of red balls in $\mathcal{C}$ (resp. $\mathcal{S}$). Thus both C_r and S_r follow a binomial distribution, and assuming they are independent, we get for $x = 0, \ldots, c$ and $y = 0, \ldots, s$

$$\beta(x, y) = \mathbb{P}\{C_r = x, S_r = y\} = \binom{c}{x}\mu^x(1-\mu)^{c-x}\binom{s}{y}\mu^y(1-\mu)^{s-y}. \tag{2}$$

The second one, that we denote δ, consists simply in starting from state $(0, 0)$, that is the state free from red balls, i.e.,

$$\delta(x, y) = \delta_{0x}\delta_{0y} \text{ with } \delta_{ij} \text{ the Kronecker delta.} \tag{3}$$

4.3.2. *Sojourn Time of both Protocols in Safe States*

We start our study by investigating for $i = 1, 2$ the distribution $T_A^{(i)}$ which counts the total number of rounds spent by $\texttt{Protocol}_i$ in the subset of safe states A before absorption. Specifically $T_A^{(1)} = \inf\{n \geq 0 \mid X_n \in B\}$. The probability mass function of T_A^1 for $k \geq 0$ is easily derived as

$$\mathbb{P}\{T_A^{(1)} = k\} = \begin{cases} \alpha_B \mathbb{1} & \text{if } k = 0 \\ \alpha_A (P_A)^{k-1}(I - P_A)\mathbb{1} & \text{if } k \geq 1 \end{cases}$$

where $\mathbb{1}$ is the column vector with all components equal to 1. The cumulative distribution function and the expectation of $T_A^{(1)}$ are respectively given by

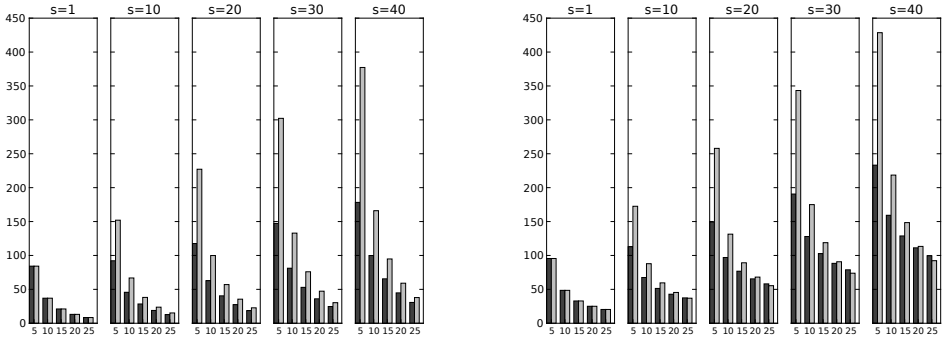
$$\mathbb{P}\{T_A^{(1)} \leq k\} = 1 - \alpha_A (P_A)^k \mathbb{1}, \text{ and } E(T_A^{(1)}) = \alpha_A (I - P_A)^{-1} \mathbb{1}. \tag{5}$$

Following the results obtained in Sericola [20], for $\texttt{Protocol}_2$, the probability mass function of $T_A^{(2)} = \sum_{n=1}^{\infty} 1_{\{X_n \in A\}}$ is given by

$$\mathbb{P}\{T_A^{(2)} = k\} = \begin{cases} 1 - v\mathbb{1} & \text{if } k = 0 \\ vR^{k-1}(I - R)\mathbb{1} & \text{if } k \geq 1 \end{cases} \tag{6}$$

where $v = \alpha_A + \alpha_C (I - P_C)^{-1} P_{CA}$ and $R = P_A + P_{AC}(I - P_C)^{-1} P_{CA}$. The cumulative distribution function and expectation of $T_A^{(2)}$ are respectively given by

$$\mathbb{P}\{T_A^{(2)} \leq k\} = 1 - vR^k \mathbb{1}, \text{ and } E(T_A^{(2)}) = v(I - R)^{-1} \mathbb{1}. \tag{7}$$



(a) $\alpha = \beta$ (see Relation (2)) (b) $\alpha = \delta$ (see Relation (3))

Fig. 5. $E(T_A^{(1)})$ (see Relation (5)) represented by dark bars and $E(T_A^{(2)})$ (see Relation (7)) represented by light grey bars as a function of s and the percentage μ of red balls. Notation $5, 10, \ldots 25$ denotes $\mu = 5\%, 10\%, \ldots, 25\%$. In all these experiments $c = 10$.

Figure 5(a) and Figure 5(b) compare the expected number of rounds run in safe states for both protocols according to the two initial distributions β and δ. In accordance with the intuition, increasing the size of the urns augments both $E(T_A^{(1)})$ and $E(T_A^{(2)})$ independently from the ratio of red balls in the bag. Similarly, for a given cluster size, augmenting the ratio of red balls in the bag decreases both $E(T_A^{(1)})$ and $E(T_A^{(2)})$. Now Figure 5(b) shows that for small values of μ $\texttt{Protocol}_2$ overpasses $\texttt{Protocol}_1$, while for larger values of μ, we observe an inverse tendency. Interpretation of this result is as follows. When the size s of $\mathcal{S}$ is equal to 1, both protocols are equivalent as illustrated in Figures 5(a) and 5(b). Now, consider the case where the size s of $\mathcal{S}$ gets larger compared to $\mathcal{C}$'s one. The probability to draw a ball from $\mathcal{S}$ tends to 1. Now as the adversary never withdraws its red balls from

any urns, the number of red balls within $\mathcal{S}$ is non decreasing. Hence, the larger μ is the faster the ratio of red balls in $\mathcal{S}$ tends to 1. With small probability, a ball from $\mathcal{C}$ is drawn. In $\texttt{Protocol}_1$ it is replaced with high probability by a red ball drawn from $\mathcal{S}$. Hence to reach a polluted state, $c'+1$ white balls have to be replaced by red ones. While with $\texttt{Protocol}_2$ the renewal of $\mathcal{C}$ reaches with high probability a polluted state in a single step. Thus, even if $\texttt{Protocol}_2$ continues to alternate for a finite number of times between safe and polluted subset of states, the total time spent in safe states is less than the one spent by $\texttt{Protocol}_1$. Note that one cannot observe such a behavior when the initial ratio of red balls in both urns is non null (see Figure 5(a)) as it takes less steps for both protocols to switch to polluted states.

4.3.3. *Alternation between Safe and Polluted States of Protocol$_2$*

A deeper investigation of $\texttt{Protocol}_2$ allows to study the duration and frequency of successive sojourn times in subsets A and C. For $n \geq 1$, we denote by $T^{(2)}_{A,n}$ (respectively $T^{(2)}_{C,n}$) the distribution of the time spent by the Markov chain X during its n-th sojourn in subset A (resp. C). The total time spent in subset A (resp. C) before reaching subset D is given by

$$T^{(2)}_A = \sum_{n=1}^{\infty} T^{(2)}_{A,n} \text{ and } T^{(2)}_C = \sum_{n=1}^{\infty} T^{(2)}_{C,n}.$$

From Sericola and Rubino [21], we have for $n \geq 1$ and $k \geq 0$

$$\mathbb{P}\{T^{(2)}_{A,n} \leq k\} = 1 - vG^{n-1}(P_A)^k \mathbb{1}, \tag{9}$$

where v is defined in Relation (6) and $G = (I - P_A)^{-1}P_{AC}(I - P_C)^{-1}P_{CA}$. Symmetrically, we have

$$\mathbb{P}\{T^{(2)}_{C,n} \leq k\} = 1 - wH^{n-1}(P_C)^k \mathbb{1}, \tag{10}$$

where $w = \alpha_C + \alpha_A(I - P_A)^{-1}P_{AC}$ and $H = (I - P_C)^{-1}P_{CA}(I - P_A)^{-1}P_{AC}$. The expectations of $T^{(2)}_{A,n}$ and $T^{(2)}_{C,n}$ are respectively given by

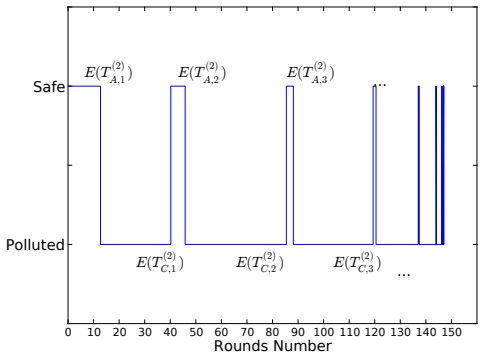
$$E(T^{(2)}_{A,n}) = vG^{n-1}(I - P_A)^{-1}\mathbb{1} \text{ and } E(T^{(2)}_{C,n}) = wH^{n-1}(I - P_C)^{-1}\mathbb{1}. \tag{11}$$

Figure 6(a) shows the expected duration and frequency of successive sojourn times in subsets A and C for $\texttt{Protocol}_2$.[a] Clearly, the protocol runs more rounds in C than it does in A. Note that for $n \geq 7$, the expected duration of the sojourn times in both A and C is already close to 0. Figure 6(b) depicts the percentage of rounds spent by $\texttt{Protocol}_2$ in safe states before absorption as a function of the size s of $\mathcal{S}$. This percentage is described by

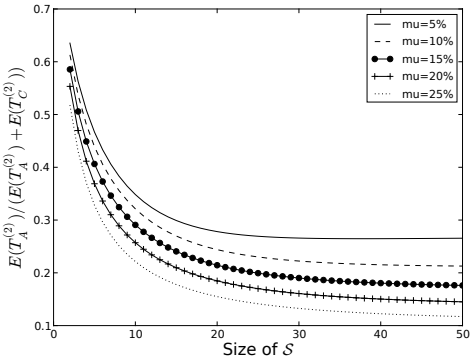
$$\frac{E(T^{(2)}_A)}{E(T^{(2)}_A) + E(T^{(2)}_C)} = \frac{v(I - R)^{-1}\mathbb{1}}{v(I - R)^{-1}\mathbb{1} + w(I - T)^{-1}\mathbb{1}} \tag{12}$$

[a]Note that if we consider the first sojourn time in the subset of safe states A of both protocols, then $\texttt{Protocol}_1$ always overpasses $\texttt{Protocol}_2$ both in expectation and with respect to their cumulative distribution functions [22])

where v and R are given in Relation (6), w is given in Relation (10), and $T = P_C + P_{CA}(I - P_A)^{-1}P_{AC}$. In accordance with the intuition, this percentage decreases with larger values of μ, and stabilizes for increasing values of s.



(a) Succession of $E(T^{(2)}_{A,n})$ and $E(T^{(2)}_{C,n})$ as a function of the rounds number. We have $\alpha = \beta$, $c = 10$, $s = 20$ and $\mu = 25\%$.



(b) $E(T^{(2)}_A)/(E(T^{(2)}_A) + E(T^{(2)}_C))$ as a function of the size s of $\mathcal{S}$. We have $\alpha = \beta$, and $c = 10$.

Fig. 6. Sojourn Times in Transient States for $\texttt{Protocol}_2$.

An aggregated view of the total time spent in transient states, i.e. A for $\texttt{Protocol}_1$, and $A \cup C$ for $\texttt{Protocol}_2$, before absorption is depicted in Figure 7. The main observation drawn from this figure is that the total time spent in transient states linearly increases with the size of $\mathcal{S}$, and thus increases logarithmically with the size of the system. On the other hand, this time is mainly spent in C, and this tendency increases with larger values of s which is confirmed by Figure 6(b).

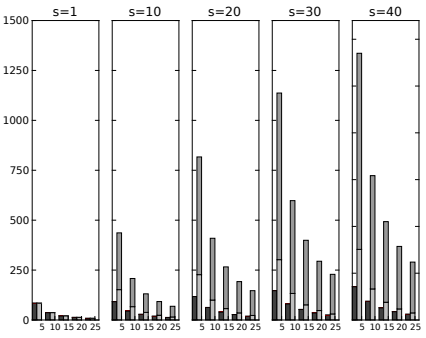


Fig. 7. $E(T^{(1)}_A)$ (Relation (5)) represented by dark colored bars, and $E(T^{(2)}_A)+E(T^{(2)}_C)$ (Relation (12)) represented by light grey and grey colored bars as a function of s and μ. In all these experiments $c = 10$.

4.3.4. *Lessons Learnt from this Comparison*

The main lessons learnt from this preliminary study is that increasing the amount of randomization of the protocol does make it necessarily more resilient to stronger adversaries. From a practical point of view this result is interesting as handling the renewal of a set of entities requires costly agreement protocols to be run among the interested parties. Typically, complexity of these protocols is in $\mathcal{O}(n^3)$ where n is the number of parties. Unfortunately by allowing malicious peers to stay indefinitely long at the same position in the overlay, both protocols are not sufficient to prevent the adversary from progressively surrounding honest peers and gaining the quorum. In the following section we show that by limiting the lifetime of malicious peers in the same cluster, safety of the whole cluster is eventually guaranteed.

5. Constraining the Adversary by Limiting the Sojourn Time in a Cluster

It has been shown [9] that structured overlays cannot survive targeted attacks if the adversary may keep sufficiently long its malicious peers at the same position in the overlay. Indeed, once malicious peers have succeeded in sitting in a focused region of the overlay, they can progressively gain the quorum by simply waiting for honest peers to leave their position, leading to the eventual isolation of honest peers. The two fundamental properties that prevent peers isolation are the guarantee that peers identifiers are random, and that peers cannot stay forever at the same position in the system [8].

From a practical point of view, implementing limited peers lifetime and unpredictable identifiers assignment, can be achieved by adding an incarnation number to the fields that appear in the peer's certificate (certificates are acquired at trustworthy Certification Authorities (CAs)), and by hashing this certificate to generate the peer's identifier. By the properties of hash functions, peers identifiers are unpredictable. The incarnation number limits the lifetime of identifiers. The current incarnation k of any peer p can be computed as $k = \lceil (t - ivt)/L \rceil$, where ivt is the initial validity time of the peer's certificate, t is the current time, and L is the length of the lifetime of each peer's incarnation. Thus, the k^{th} incarnation of peer p expires when its local clock reads $ivt + kL$. At this time p must rejoin the system using its $(k+1)^{th}$ incarnation. This is discussed in details in [22].

Coming back to our analysis, we model the constraint on the adversary by preventing the adversary from keeping its red balls in both urns infinitely long, so that randomness among red and white balls is continuously preserved. As previously, we investigate both protocols presented in Figure 1. It is not difficult to see that both protocols alternate between the subset of safe states $A = \{(x, y) \mid 0 \leq x \leq c',\ 0 \leq y \leq s\}$, and the subset of polluted ones $B = \{(x, y) \mid c' < x \leq c,\ 0 \leq y \leq s\}$ for an infinite number of rounds. Both subsets A and B are transient, preventing the adversary from ever winning any of the two protocols. Following the decomposition

of $\Omega = A \cup B$, we partition matrix P and the initial probability vector α by writing

$$P = \begin{pmatrix} P_A & P_{AB} \\ P_{BA} & P_B \end{pmatrix} \text{ and } \alpha = (\alpha_A \ \ \alpha_B).$$

5.1. *Protocol*$_1$

By proceeding exactly as in Section 4, we derive the transition probability matrix P for $\texttt{Protocol}_1$. That is, for all $x \in \{0, \ldots, c\}$ and $y \in \{0, \ldots, s\}$, we have

$$\begin{aligned}
p_{(x,y),(x,y)} &= \frac{xy + (c(s-y) - xs)(1-\mu)}{(c+s)s} + \frac{y\mu + (s-y)(1-\mu)}{c+s} \\
p_{(x,y),(x,y-1)} &= \frac{(x+s)y}{(c+s)s}(1-\mu) \text{ for } y \geq 1 \\
p_{(x,y),(x,y+1)} &= \left(\frac{c-x+s}{c+s}\right)\left(\frac{s-y}{s}\right)\mu \text{ for } y \leq s-1 \\
p_{(x,y),(x+1,y-1)} &= \frac{(c-x)y}{(c+s)s}(1-\mu) \text{ for } x \leq c-1 \text{ and } y \geq 1 \\
p_{(x,y),(x+1,y)} &= \frac{(c-x)y}{(c+s)s}\mu \text{ for } x \leq c-1 \\
p_{(x,y),(x-1,y)} &= \frac{x(s-y)}{(c+s)s}(1-\mu) \text{ for } x \geq 1 \\
p_{(x,y),(x-1,y+1)} &= \frac{x(s-y)}{(c+s)s}\mu \text{ for } x \geq 1 \text{ and } y \leq s-1.
\end{aligned} \tag{13}$$

In all other cases, transition probabilities are null.

5.2. *Protocol*$_2$

Similarly for all $x \in \{0, \ldots, c\}$ and $y \in \{0, \ldots, s\}$ the entries of P for $\texttt{Protocol}_2$ are given by

$$\begin{aligned}
p_{(x,y),(x,y)} &= \frac{xq(x,x+y-1)\mu}{c+s} + \frac{(c-x)q(x,x+y)(1-\mu)}{c+s} + \frac{y\mu + (s-y)(1-\mu)}{c+s} \\
p_{(x,y),(x,y-1)} &= \frac{x}{c+s}q(x,x+y-1)(1-\mu) + \frac{y}{c+s}(1-\mu) \text{ for } y \geq 1 \\
p_{(x,y),(x,y+1)} &= \frac{c-x}{c+s}q(x,x+y)\mu + \frac{s-y}{c+s}\mu \text{ for } y \leq s-1 \\
p_{(x,y),(k,x+y-k-1)} &= \frac{x}{c+s}q(k,x+y-1)(1-\mu) \\
&\quad \text{for } \max(0, x+y-1-s) \leq k \leq \min(c, x+y-1) \text{ and } k \neq x \\
p_{(x,y),(k,x+y-k)} &= \frac{x}{c+s}q(k,x+y-1)\mu + \frac{c-x}{c+s}q(k,x+y)(1-\mu) \\
&\quad \text{for } \max(0, x+y-s) \leq k \leq \min(c, x+y-1) \text{ and } k \neq x \\
p_{(x,y),(k,x+y-k+1)} &= \frac{c-x}{c+s}q(k,x+y)\mu \\
&\quad \text{for } \max(0, x+y+1-s) \leq k \leq \min(c, x+y) \text{ and } k \neq x,
\end{aligned} \tag{14}$$

where $q(z, x+y)$ is given by Relation (1). In all other cases, transition probabilities are null.

5.3. *Comparison of both Protocols in a Constrained Adversarial Environment*

By proceeding as in Section 4.3.3, we investigate the behavior of both protocols during their successive n-th sojourn time in A and B. Each Markov chain X is irreducible and aperiodic since at least one state has a transition to itself. For $n \geq 1$, we denote by respectively $T_{A,n}^{(i)}$ and $T_{B,n}^{(i)}$ the time spent by Markov chain X associated with $\texttt{Protocol}_i$ during its n-th sojourn in respectively subsets A and B. Expectations of $T_{A,n}^{(i)}$ and $T_{B,n}^{(i)}$ are respectively given for $i = 1, 2$ by

$$E(T_{A,n}^{(i)}) = vG^{n-1}(I - P_A)^{-1}\mathbb{1} \text{ and } E(T_{B,n}^{(i)}) = wH^{n-1}(I - P_B)^{-1}\mathbb{1} \tag{15}$$

where $v = \alpha_A + \alpha_B(I - P_B)^{-1}P_{BA}$, $G = (I - P_A)^{-1}P_{AB}(I - P_B)^{-1}P_{BA}$, $w = \alpha_B + \alpha_A(I - P_A)^{-1}P_{AB}$ and $H = (I - P_B)^{-1}P_{BA}(I - P_A)^{-1}P_{AB}$.

For both protocols, the Markov chain X is finite, irreducible and aperiodic so the stationary distribution exists and is unique. We denote by π the stationary distribution of the Markov chain X. The row vector π is thus the unique solution to the linear system $\pi P = \pi$ and $\pi\mathbb{1} = 1$. As we did for row vector α, we partition vector π according to the partition $\Omega = A \cup B$ by writing $\pi = (\pi_A \quad \pi_B)$, where sub-vector π_A (resp. π_B) contains the stationary probabilities of states of A (resp. B). The mean percentage of time spent in subset A during the j-th sojourn is equal for $\texttt{Protocol}_i$ (i=1,2) to $E(T_{A,j}^{(i)})/(E(T_{A,j}^{(i)}) + E(T_{B,j}^{(i)}))$. By Cesàro lemma,

$$\lim_{n\to\infty} \frac{\sum_{j=1}^{n} E(T_{A,j}^{(i)})}{\sum_{j=1}^{n}(E(T_{A,j}^{(i)}) + E(T_{B,j}^{(i)}))} = \lim_{n\to\infty} \frac{E(T_{A,n}^{(i)})}{E(T_{A,n}^{(i)}) + E(T_{B,n}^{(i)})} = \pi_A\mathbb{1} \tag{16}$$

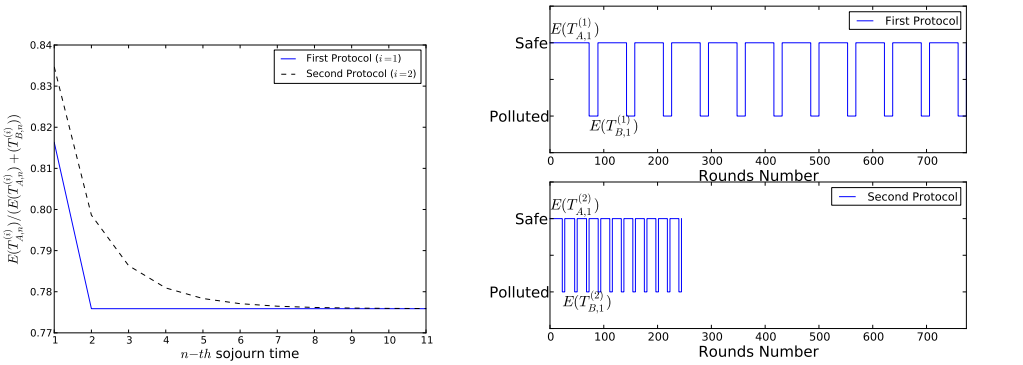
and the first hitting time to reach subset B is given, for every $k \geq 0$, by

$$\mathbb{P}\{T_{A,1}^{(i)} \leq k\} = 1 - \alpha_A(P_A)^k\mathbb{1}.$$

Figures 8(a) and 8(b) show the behavior of the Markov chains associated with both protocols during their first 11 sojourn times in both A and B. First, for $\mu = 25\%$ both protocols spend more than 3/4 of their time in safe states and their convergence speed to $\pi_A\mathbb{1}$ is very fast (convergence is reached in a single step for $\texttt{Protocol}_1$ and in 6 steps for $\texttt{Protocol}_2$) as shown in Figure 8(a). Figure 8(b) shows that the frequency at which safe and polluted states alternate is 3 times lower for $\texttt{Protocol}_1$ than for $\texttt{Protocol}_2$. From a practical point of view this is interesting as it makes $\texttt{Protocol}_1$ more adapted to dynamic environment compared to $\texttt{Protocol}_2$ (i.e., $\texttt{Protocol}_1$ handles a higher number of connections and disconnections before switching to a polluted state than $\texttt{Protocol}_2$ does).

Theorem 1. *For both protocols, the stationary distribution π is equal to β, i.e. for all $x = 0, \ldots, c$ and $y = 0, \ldots, s$, we have*

$$\lim_{n\longrightarrow\infty} \mathbb{P}\{X_n = (x, y)\} = \beta(x, y) \text{ with } \beta(x, y) \text{ given by relation (2).}$$



(a) $E(T_{A,n}^{(i)})/(E(T_{A,n}^{(i)}) + E(T_{B,n}^{(i)}))$ as a function of the n-th sojourn time (Relation (15)). We have $\alpha = \beta$, $c = 10$, $s = 20$ and $\mu = 25\%$.

(b) Succession of $E(T_{A,n}^{(i)})$ and $E(T_{B,n}^{(i)})$ as a function of the rounds number. We have $\alpha = \beta$, $c = 10$, $s = 20$ and $\mu = 25\%$.

Fig. 8. Comparison of sojourn times in safe and polluted states.

Proof. For space reason, the proof appears in [22]. □

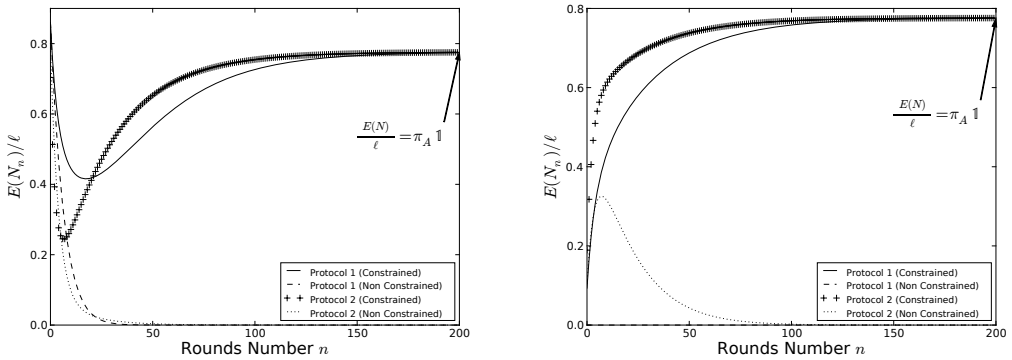
Theorem 1 is quite interesting as it shows that the steady state of the system which is described by the stationary distribution π is exactly the same for both protocols, and that this distribution is equal to distribution β (independently from the initial distribution). At a first glance, we could guess that this phenomenon is due to the fact that the Markov chain X is the tensor product of two independent Markov chains. However, this is clearly not the case as the behavior of red balls in $\mathcal{C}$ depends on the behavior of red balls in $\mathcal{S}$. This holds for both protocols. The stationary availability of the system defined by the long run probability to be in safe states is denoted by P_{safe} and is given by

$$P_{\text{safe}} = \pi_A \mathbb{1} = \sum_{x=0}^{c'} \binom{c}{x} \mu^x (1-\mu)^{c-x}.$$

This probability can also be interpreted as the long run proportion of time spent in safe states. Note that the stationary distribution does not depend on the size of $\mathcal{S}$.

6. Robustness of the Overlay Network

We now show that by inducing global churn, we can preserve the safety of the system. We consider that we have ℓ identical and independent Markov chains $X^{(1)}, \ldots, X^{(\ell)}$ on the same state space $\Omega = A \cup B$, with initial probability distribution α and transition probability matrix P. Each Markov chain $X^{(i)}$ models a particular cluster of peers and, for $n \geq 0$, N_n represents the number of safe clusters after the n-th round, i.e. the number of Markov chains being in subset A after the



(a) $C_r = 3$ and $S_r = 12$, i.e. $X_0 = (3, 12)$ (b) $C_r = 4$ and $S_r = 6$, i.e. $X_0 = (4, 6)$

Fig. 9. Percentage of the expected number of safe clusters (relation (18)) as a function of the rounds number n for both protocols, for two different initial states X_0 (Section 4.3.1). In these experiments, $\ell = 100$, $c = 10$, $s = 20$, and $\mu = 25\%$.

n-th transition has been triggered, defined by $N_n = \sum_{i=1}^{\ell} 1_{\{X_n^{(i)} \in A\}}$. The ℓ Markov chains being identical and independent, N_n has a binomial distribution, that is, for $k = 0, \ldots, \ell$

$$\mathbb{P}\{N_n = k\} = \binom{\ell}{k} (\alpha P^n \mathbb{1})^k (1 - \alpha P^n \mathbb{1})^{\ell - k} \text{ and } E(N_n) = \ell \alpha P^n \mathbb{1}. \tag{18}$$

If N denotes the stationary number of safe clusters, we have

$$E(N) = \begin{cases} \ell \pi_A \mathbb{1} & \text{for a constrained adversary} \\ 0 & \text{for a non constrained adversary} \end{cases} \tag{19}$$

These results are illustrated in Figure 9. The main observation is that with a constrained adversary, the expected number of safe clusters for both protocols tends to the same limit $\ell \pi_A \mathbb{1}$ whatever the amount of initial pollution, while with a non constrained adversary all clusters get eventually polluted. This clearly shows that by limiting the time spent by peers at the same position in the overlay, targeted attacks are tolerated.

7. Practical Significance of the Results

In this section, we evaluate the practical significance of our results when instantiated with real traces. Traces we are referring to have been collected in the Overnet file sharing network and analyzed in [23]. The main reasons for using these traces is that Overnet peers are identified by permanent IDs which allows to observe fine-grained behavior of active peers in the overlay. Traces which were collected for a period of 15 days have shown that every day between $70,000$ and $90,000$ peers IDs are present in the overlay. It has been registered that each peer triggers in average

6.4 join and leave events per day, that is in average each peer joins the overlay every 7.5 hours. Thus, for an average population of 80,000 peers, this makes 512 970 join and leave events per day. In our context, an average population of $U = 80,000$ peers leads to around $\ell = U/\lfloor \log_2 U \rfloor = 5,000$ clusters each populated by $\lfloor \log_2 U \rfloor = 16$ peers, and thus to around $U/\ell \times 3.2 = 52$ leave-join rounds per day per cluster. Thus, in a presence of a non-constrained adversary, extrapolation of Figure 7 shows that for $\mu = 5\%$, $c = 10$ and for $s = 6$, it takes one day and a half (i.e., 85 rounds) for the adversary to definitively pollute a targeted cluster when $\texttt{Protocol}_1$ is run while it takes 5 days (i.e., 260 rounds) when $\texttt{Protocol}_2$ is used. Moreover, in the presence of a constrained adversary, one can derive from our study the average lifetime that must be imposed on malicious peers to prevent pollution. This comes down to compute the average frequency at which a cluster becomes polluted which is given by $1/(\alpha_A P_{AB} \mathbb{1})$. Coming back to the traces, for $c = 10$, $s = 6$ and $\mu = 5\%$, peers must leave their cluster every 104 days (5455 rounds) when $\texttt{Protocol}_1$ is run and every 41 days (i.e., 2152 rounds) when $\texttt{Protocol}_2$ is used to prevent pollution. Now, when $\mu = 25\%$, malicious peers need to leave every 20 hours (i.e., 45 rounds) with $\texttt{Protocol}_1$ and every 7 hours (i.e., 2152 rounds) with $\texttt{Protocol}_2$. Clearly these frequencies remain fully compatible with the rate at which honest peers have been observed to leave the system (i.e., every $24/3,2 = 7.5$ hours in average). This clearly show the practicability of the induced churn approach in large scale systems.

8. Conclusion

In this paper, we have proposed a mechanism that enables the enforcement of limited peers lifetime compliant with DHT-based overlays specificities. We have investigated the long run behavior of several adversarial strategies. Our analysis has demonstrated that an adversary can easily subvert a cluster-based overlay by simply never triggering leave operations. We have shown that when peers have to regularly leave the system, a stationary regime where the ratio of malicious peers is bounded is eventually reached. Finally, we have shown that this induced churn is fully compatible with the natural churn observed in peer-to-peer systems, which makes our approach definitively adapted to large scale and open systems.

References

[1] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach, "Secure routing for structured peer-to-peer overlay networks," in *Proceedings of the Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[2] A. Singh, T. Ngan, P. Drushel, and D. Wallach, "Eclipse attacks on overlay networks: Threats and defenses," in *Proceedings of the Conference on Computer Communications (INFOCOM)*, 2006.

[3] E. Sit and R. Morris, "Security considerations for peer-to-peer distributed hash tables," in *Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.

[4] A. Fiat, J. Saia, and M. Young, "Making chord robust to byzantine attacks," in *Proceedings of the Annual European Symposium on Algorithms (AESA)*, 2005.

[5] I. Baumgart and S. Mies, "S/kademlia: A practicable approach towards secure key-based routing," in *Proceedings of the International Conference on Parallel and Distributed Systems (ICPADS)*, 2007.

[6] E. Anceaume, F. Brasileiro, R. Ludinard, and A. Ravoaja, "PeerCube: an hypercube-based p2p overlay robust against collusion and churn," in *Proceedings of the International Conference on Self-Adaptive and Self-Organizing Systems (SASO)*, 2008.

[7] M. Srivatsa and L. Liu, "Vulnerabilities and security threats in structured peer-to-peer systems: A quantitiative analysis," in *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC)*, 2004.

[8] B. Awerbuch and C. Scheideler, "Towards scalable and robust overay networks," in *Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS)*, 2007.

[9] ——, "Group spreading: A protocol for provably secure distributed name service," in *Proceedings of the 31rst International Colloquium on Automata, Languages and Programming (ICALP)*, 2004.

[10] P. B. Godfrey, S. Shenker, and I. Stoica, "Minimizing churn in distributed systems," in *Proceedings of the ACM SIGCOMM*, 2006.

[11] T. Locher, S. Schmid, and R. Wattenhofer, "eQuus: A provably robust and locality-aware peer-to-peer system," in *Proceedings of the International Conference on Peer-to-Peer Computing (P2P)*, 2006.

[12] E. Bortnikov, M. Gurevich, I. Keidar, G. Kliot, and A. Shraer, "Brahms: Byzantine resilient random membership sampling," *Computer Networks*, vol. 53, no. 13, pp. 2340–2359, 2009.

[13] C. G. Plaxton, R. Rajaraman, and A. W. Richa, "Accessing nearby copies of replicated objects in a distributed environment," in *Proceedings of the 9th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, 1997.

[14] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proceedings of the ACM SIGCOMM*, 2001.

[15] I. Stoica, D. Liben-Nowell, R. Morris, D. Karger, F. Dabek, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proceedings of the ACM SIGCOMM*, 2001.

[16] P. Druschel and A. Rowstron, "Past: A large-scale, persistent peer-to-peer storage utility," in *Proceedings of the Workshop on Hot Topics in Operating Systems (HotOS)*, 2001.

[17] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the XOR metric," in *Proceedings for the International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.

[18] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, 1982.

[19] J. Douceur, "The sybil attack," in *Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.

[20] B. Sericola, "Closed form solution for the distribution of the total time spent in a subset of states of a Markov process during a finite observation period," *Journal of Applied Probability*, vol. 27, no. 3, pp. 713–719, 1990.

[21] B. Sericola and G. Rubino, "Sojourn times in Markov processes," *Journal of Applied Probability*, vol. 26, no. 4, pp. 744–756, 1989.

[22] E. Anceaume, F. Brasileiro, R. Ludinard, B. Sericola, and F. Tronel, "Dependability evaluation of cluster-based distributed systems," IRISA, Tech. Rep. 1947, 2010.

[23] R. Bhagwan, S. Savage, and G. M. Voelker, "Understanding availability," in *Proceedings of the International Workshop in Peer-to-Peer Systems (IPTPS)*, 2003.